REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY (Leave blank)	2. REPORT DATE	3. REPORT TYPE AND DATES COVERED
		FINAL TECHNICAL 01 AUG 91 TO 30 SEP 94

4. TITLE AND SUBTITLE
FOUNDATIONS OF TECHNOLOGY FOR CONSTRUCTING HIGHLY RELIABLE DISTRIBUTED REALTIME SYSTEMS

5. FUNDING NUMBERS
61102F
2304/A2

6. AUTHOR(S)
PROFESSOR DAVID C. LUCKHAM

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
DEPT OF ELECTRICAL ENGINEERING, COMPUTER SYSTEMS LAB
STANFORD UNIVERSITY
STANFORD CA 94305-4125

8. PERFORMING ORGANIZATION REPORT NUMBER
AFOSR-TR-95-0235

9. SPONSORING MONITORING AGENCY NAME(S) AND ADDRESS(ES)
DEPT OF THE AIR FORCE
AIR FORCE OFFICE OF SCIENTIFIC RESEARCH (AFMC)
BOLLING AFB DC 20332-0001

10. SPONSORING MONITORING AGENCY REPORT NUMBER
AFOSR-91-0354

11. SUPPLEMENTARY NOTES

12a. DISTRIBUTION AVAILABILITY STATEMENT
APPROVED FOR PUBLIC RELEASE: DISTRIBUTION IS UNLIMITED.

12b. DISTRIBUTION CODE

13. ABSTRACT (Maximum 200 words)

Under grant AFOSR-91-0354 we have investigated event-based specification and constraint language extensions of our Rapide prototyping language. We have also investigated testing methods and tools for detecting constraint violations in simulations of distributed time-sensitive avionics systems and control systems. Rapide models the behavior of a distributed system by generating causal event simulations. A causal event simulation is a timed poset (partially ordered set of events with timing). Dependencies between events as well as their timing are captured in the poset execution model, thus providing a more detailed and precise picture of the behavior of a realtime, distributed system than current simulation technology based upon sequential traces of events. Posets allow more powerful constraint specifications than traces, e.g., asynchronous behavior. This work has developed (i) basic algorithms for implementing poset computations. (ii) a constraint language for specifying behavior in terms of posets, and (iii) automatable algorithms and tool-set for detecting constraint violations in posets. To establish the feasibility of scaling this simulation technology to practical avionics examples, we have applied the technology to developing high level systems architectures of avionics systems. We have also applied constraint monitoring of the avionics simulations to detect design errors. The avionics systems studied include the IBM ADAGE helicopter avionics system architecture, and a high level architecture of the Boeing DARTS system for building flight simulators.

14. SUBJECT TERMS

15. NUMBER OF PAGES

16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
UNCLASSIFIED	UNCLASSIFIED	UNCLASSIFIED	UNLIMITED

Final Report under AFOSR Grant–91–0354A

Foundations of Technology for Constructing Highly Reliable Distributed Realtime Systems

August 1, 1991 – September 30, 1994

Principal Investigator: David C. Luckham

1 Abstract

Under AFOSR grant 91-0354A we have investigated event-based specification languages that can be positioned as constraint language extensions of our Rapide prototyping language. We have also investigated testing methods based upon tools for detecting constraint violations in simulations of distributed time-sensitive avionics systems and control systems. Rapide models the behavior of a distributed system by generating causal event simulations of such systems. A causal event simulation is a *timed poset* (partially ordered set of events with timing). Dependencies between events as well as their timing are captured in the poset execution model, thus providing a more detailed and precise picture of the behavior of a realtime, distributed system than current simulation technology based upon sequential traces of events. Posets allow more powerful constraint specifications of, e.g., asynchronous behavior, than traces. This work has developed *(i)* basic algorithms for implementing poset computations, *(ii)* a constraint language for specifying behavior in terms of posets, and *(iii)* automatable algorithms and tool-set for detecting constraint violations in posets produced by Rapide simulations. To establish the feasibility of scaling this simulation technology to practical avionics examples, we have applied it to developing high level system architectures of avionics systems in Rapide, and applying constraint monitoring to simulations of these architectures to detect design errors. The avionics systems studied include IBM ADAGE helicopter avionics system architecture, and a high level architecture of the Boeing DARTS system for building flight simulators.

2 Introduction

Evolutionary prototyping is an emerging technology for integrating rigorous systems design methods with highly accurate behavioral analysis methods throughout the system development process. Rigorous design methods can be founded upon, for example, the use of formal specifications, which by and large are not used in current practice. Accurate analysis methods require a foundational

basis of more detailed and precise models of timed distributed computation than are currently used. But above all, evolutionary prototyping is focussed on the need to integrate design and analysis throughout the development of a system in a very tightly coupled, fine-grain process. This is in direct contrast with current systems development technology which separates design and analysis into distinct phases, often resulting in large and costly redesign efforts.

To ensure relevance and applicability, our research has developed constraint-based specification languages and analysis methods to support current, on-going projects in prototyping distributed, time-sensitive systems. The subject projects chosen to study applicability of our work include the Stanford/TRW ARPA Prototech project, avionics and control architectures from two of the ARPA DSSA projects, and the Boeing DARTS system.

The Rapide prototyping language design is based upon the model of a distributed computation as a partially ordered set of events with timing measures (*Timed Poset*). This is the Rapide execution model adopted in the Stanford/TRW Prototech project [1]. This model has already been shown to provide significantly more information for analysis of system behavior than the commonly used linear trace model. Also, our work on this project in analysis tools [9] shows that this information, although complex, can be organized for processing by automated tools such as error checkers and graphical depiction tools.

This research effort under AFOSR grant Grant–91–0354A has accomplished the following goals:

1. development of basic algorithms for implementing poset computations,
2. design of a constraint language for specifying system behavior in terms of posets. This constraint language has the properties that
   - the constraints are powerful enough to express many critical properties of avionics system architectures, including communication protocols and timing behavior,
   - violations of the constraints in poset simulations can be detected by simple algorithms at runtime.
3. implementation of a pilot tool-set, extending the Rapide simulator tools, for detecting constraint violations in posets produced by Rapide simulations.
4. demonstration of feasibility of scaling this simulation technology to practical avionics examples. We have represented high level functional architectures of avionics systems in Rapide and applied constraint monitoring to Rapide simulations of these architectures to detect design errors. The avionics systems studied include
   - IBM ADAGE helicopter avionics system architecture, and
   - a high level architecture of the Boeing DARTS system for building flight simulators.

These accomplishments over the lifetime of the AFOSR grant establish that we have achieved our original goals of the project, which were to : *(i)* develop accurate simulations of distributed time sensitive computations, perhaps improving upon the timed poset model, and algorithms for constructing such simulations by executing architectural models, *(ii)* design a language for specifying systems in terms of the timed poset model of their behaviors, and *(iii)* implement and test the scalability of analytical tools for applying constraint specifications to analysis of avionics system architectures expressed in Rapide.

3 Context of the Overall Effort

RAPIDE is an executable ADL. Its purpose is *(i)* to study and develop a foundational theory of distributed system architectures, and *(ii)* to design language constructs for expressing architecture in an executable form to enable early prototyping of systems designs. RAPIDE supports execution and analysis of system architectures using the timed poset execution model for distributed/concurrent systems. The advantage of timed posets over timed linear traces is that they capture not only the time at which events happen, but also dependency between events (i.e., which events *caused* which other events to happen, and which events happened independently). This model is actually constructed when a prototype of a system is executed in the RAPIDE prototyping language [1]. Algorithms to do this have been implemented that optimize techniques due to Fidge [2]. Similar models can be implemented for Ada systems following our previous work on TSL.

The design of RAPIDE uses event-pattern concepts from the TSL (specification language for Ada tasking) and VAL (specification language for VHDL) specification languages which were developed under our previous AFOSR grants. Over the period of this AFOSR grant we have developed analysis algorithms for checking event pattern constraints for consistency with timed poset executions. This provides the foundation for an automated capability to check the behavior of prototype distributed systems at runtime for violations of specifications.

3.1 Distributed Execution Models

Over the past four years our work has demonstrated the feasibility of implementing languages that generate posets.

A number of open questions about models of distributed realtime computations remain. These have to do with the possibility of constructing even more accurate models, and the scalability of current modelling technology to handle very large systems. Our work under this AFOSR grant was aimed at investigating such issues, for example :

- **Implementation algorithms for scalability to 10K concurrency.**
  Fundamental questions of scalability of the current implementation algorithm from a capability to deal with systems involving a hundred concurrent threads of control to systems with many thousands of threads remain open. We have improved existing algorithms to handle 2K events per second in models of highly concurrent systems.

- **Models of widely distributed Local Time**
  All applications of timed poset models that we have investigated so far model systems containing a single global clock. All components of such systems, no matter how far apart, can read the same clock. Although our current timed posets can model widely distributed timed systems, such as the Internet where each net node has its own local clock, we have not yet studied models that make essential use of this facility. Examples to be studied are satellite communication systems. We have not yet been able to initiate a research effort to study this problem in detail, although the RAPIDE LRMs do provide a design for distributed local clocks.

3.2 Constraint-Based Specification Language Design

Specifications of a highly parallel system can be given in terms of the timed poset model. To do this, the required behaviors (timed posets) of the system are specified by means of constraints bounding the allowable variations of behavior. In general, such specifications define patterns of dependency, concurrency, synchronization, and timing between the sets of events that the system produces when it executes. Quite often, the specified activity is dependent on the state of the system, and also it is common to have many events that are irrelevant to a particular behavior. Accurate specifications must therefore refer to patterns of events, state context, and be capable of hiding (or abstracting) unnecessary details.

A critical step in the development of RAPIDE technology is the design of a powerful abstract interface specification language based on poset executions. This is important in providing a capability to specify interfaces of components in a large distributed system, particularly enabling the application of object-oriented methods to such specifications.

During the period of this AFOSR grant we able to design a constraint-based specification language to fit with RAPIDE that provided several of the desired features : *(i)* relational operators (e.g., immediately relevant, dependent, parallel, time relation), *(ii)* subcomputation templates (e.g., patterns of events), *(iii)* state context (e.g., use of abstract data types as state specifications), *(iv)* hiding (e.g., filters for excluding irrelevant events).

However, only a subset of this language has been implemented in the constraint violation detection tools. This subset is indeed powerful enough to express many system specifications. But many practical issues surrounding complexity of checking algorithms for constraint on posets remain to be investigated.

3.3 Specification of Distributed Timed Computations

Specifying timing behavior has proven to be one of the most difficult aspects in the description of real-time systems. The different views of time espoused by systems to date can be classified into at least three groups. Although the placement of a particular language into one group is sometimes debatable, the classes are still useful for organizing the various views of time.

We have already developed a simple global timing model for a preliminary implementation of RAPIDE. This allows posets of events to be produced by prototypes in which the events are assigned timestamps and timing intervals. Thus, events that overlap in time can be represented in timed poset models.

Some of the goals we planned to attempt under the present AFOSR investigation into timing specifications of distributed time-critical systems were only partially attempted due to time and resource limitations. Some remaining issues are :

1. Develop a set of benchmark timing examples which capture the different facets of time in the modelling of time-critical distributed systems.
2. Evaluate different timing specification languages for applicability to modeling distributed systems with multiple clocks.
3. Formulate a timing specification language, possibly based on an existing language, suitable for modeling distributed systems with multiple clocks.

3.4 Constraint Checking Algorithms

We have implemented constraint checkers for a constraint-based RAPIDE Specification Language. These tools allow runtime checking of event pattern constraint on posets. They have been used in conjunction with RAPIDE analysis tools such as the Poset Browser, and have been tested on many examples developed both within and outside our research group. These examples include the IBM avionics system which is one of the ARPA DSSA benchmark system architectures. Much further research into practical runtime violation detection algorithms for even more powerful constraints is still needed.

4 Results Achieved and Technology Impact

Previous annual reports have described the progress and impact of this project in prior years. During the past year, the main thrusts of this AFOSR project have been:

- *(i)* integration of a constraint language into the RAPIDE simulation language.
- *(ii)* completion of the constraint checking tool, and its integration with the RAPIDE compiler and runtime simulation scheduler.
- *(iii)* completion of a feasibility study to apply constraint checking to RAPIDE simulations of industrial-scale avionics problems, including ADAGE, and DARTS, as well as to other systems including AEGIS, SPARC V9, and X/Open DTP [11, 5].

This work carried out under our AFOSR grant has had a critical impact on our RAPIDE technology development and studies aimed at establishing scalability to industrial size systems and finally, transition to industry. The constraint language checker is now an integral part of the current RAPIDE simulation and analysis toolset, which is being implemented under ARPA contracts and tested by TRW and other industry companies. The tools are also been used by outside users and by students in a course on design and prototyping. The range of applications that have been the subject of various RAPIDE studies include :

- two avionics systems (ADAGE and DARTS),
- a model of the safety critical subsystem of an air traffic control system,
- a shared memory multiprocessor [10],
- the super SPARC CPU architecture,
- distributed transaction management systems such as X/Open,
- the Navy's AEGIS weapon system

We have given accounts of much of this work in previous annual reports, particularly ADAGE. However, the application to DARTS was only initiated during FY94, so we give a short overview of this work in the following section.

5 Application to the DARTS System

In the following we give a brief summary of our most recent application to an industry avionics system. This work is ongoing at the time the AFOSR Grant terminated, but had been taken far enough to provide good evidence of RAPIDE applicability to flight simulator system architectures.

5.1 Background

DARTS, Domain Architecture for Reuse in Training Systems, is the software architecture/system architecture for Air Vehicle Training Systems (AVTS) developed by Boeing's Defense and Space Group.

The English description of DARTS, in 12 volumes of documents, is ambiguous, inconsistent, flat and difficult to understand, which hinders the cost/performance study in application engineering.

Boeing desired an executable model of DARTS that

1. Provides a compressed presentation of DARTS system-level design out of the presently existing 12 volumes of documentation.
2. Supports trade studies of DARTS at system-level by showing cost functions and performance measurements for different designs/configurations/work-load-distributions.
3. Helps answer two system-level questions very early in application engineering process: Will this design work ? How does this design compare with other designs ?

5.2 Modeling DARTS in Rapide

Rapide is an executable architecture definition language, developed by Stanford's Program Verification and Analysis Group. Rapide's unique advanced features enable design engineers to define their target system's architecture at different levels of abstraction, to execute the defined architecture, and to check and analyze results produced by the execution.

Rapide modeling of DARTS precisely addresses the problems of ambiguity, inconsistence, flatness and difficulty in understanding with the DARTS documents.

The Rapide model of DARTS that was developed during this effort provides

1. Executable simulation of DARTS at the system-level (i.e., Fundamental MSS Partitioning level, in Boeing's jargon).
2. Display of event flow and timing relationship of DARTS' Segments, Subsystem Controller and Components in simulation execution.
3. System-level performance measurements with different designs/configurations.
4. Assistance in cost estimation. Once the performance of different designs is obtained, calculating the cost/performance would be a simple optimization problem with tools such as Mathematica.

By "system-level" simulation, we mean executing the DARTS model at abstract functional level, i.e., the functions are abstracted such that important/relevant events and dependences can be seen without using the full implementation of the functions.

5.3 Achievements to date

We have approached DARTS modeling in a top-down hierarchical manner.

We have built our first Rapide model of DARTS at the fundamental MSS partitioning level, which is the very high level of abstraction of DARTS. This model includes (1) the Global Simulation Control, the MSS components and the communication mechanism, (2) the interface functions of Global Simulation Control, and the MSS components.

We have executed the DARTS model in Rapide. The execution simulates 3 phases of high-level DARTS system operation: initialization, frame transmission and completion. The execution has provided event flow and dependences in visual form, which will be used for analysis and verification.

5.4 Continuation of the DARTS Modelling Study

At the next level, our initial model will be expanded to including protocols, timing estimates, throughput estimates, and other details.

At the third level, our model at the second level will be further expanded and may provide (1) high-level description of all functions in Module Executives, Segment Executives, Subsystem Controllers and Components, (2) the protocols of communications between Module Executives, Segment Executives, Subsystem Controllers and Components, (3) Sample but complete simulation frames, and (4) a scenario that explains how all segments react to some typical situations (encoded in frames) and collaborate to carry out tasks.

6 Conclusions

This work carried out under AFOSR grant Grant–91–0354A has had a critical impact on our RAPIDE technology development and studies aimed at establishing scalability to industrial size systems and finally, transition to industry. The constraint language checker is now an integral part of the current RAPIDE simulation and analysis toolset, which is being implemented under ARPA contracts and tested by TRW and other industry companies. The tools are also been used by outside users and by students in a course on design and prototyping. The range of applications that have been the subject of various RAPIDE studies include :

- two avionics systems (ADAGE and DARTS),
- a model of the safety critical subsystem of an air traffic control system,
- a shared memory multiprocessor [10],
- the super SPARC CPU architecture,
- distributed transaction management systems such as X/Open,
- the Navy's AEGIS weapon system

7 List of Publications Under AFOSR grant Grant–91–0354A

1. Luckham, D.C., Vera, J., Bryan, D., Augustin, L.M., and Belz, F., *"Partial Orderings of Event Sets and Their Application to Prototyping Concurrent Timed Systems"*, in proceedings of DARPA Software Technology Conference, Los Angeles, California, pages 443-457, April 28-30, 1992.

2. Mitchell, J.C., Katiyar, D., Luckham, D.C., Madhav, N., Meldal, S., and Sankar, S., *"Subtyping, Assignment and Cloning in a Concurrent Object-Oriented Language."*, in proceedings of DARPA Software Technology Conference, Los Angeles, California, pages 458-470, April 28-30, 1992.

3. Luckham, D.C., Sankar, S., Mann, W., and Goyal, A., *"Anna – A Language For Specifying Ada Programs"*, in proceedings of DARPA Software Technology Conference, Los Angeles, California, pages 498-501, April 28-30, 1992.

4. Luckham, D.C., Vera, J., Bryan, D., Augustin, L., and Belz, F., *"Partial orderings of event sets and their application to prototyping concurrent timed systems"*, Computer Systems Laboratory Technical Report CSL–TR–92–515, (PAVG No. 59), April, 1992.

5. **1992 ACM/IEEE Best Paper Award**,
   Gennart, Benoit A., and Luckham, David C., *"Validating Discrete Event Simulations Using Event Pattern Mappings"*, Proceedings of the 29th ACM/IEEE Design Automation Conference, Anaheim, California, pages 414-419, June 8 – 12, 1992.

6. Luckham, D.C., Vera, J., Bryan D., Augustin L.M., and Belz F., *"Partial Orderings of Event Sets and Their Application to Prototyping Concurrent Timed Systems"*, Journal of Systems and Software, Elsevier Press, New York, special issue on applying specification, verification, and validation techniques to industrial software systems, Vol 21(3) pp.251-265, June, 1993.

7. Madhav, N., *"Correctness and Error Detection in a Model of Distributed Program Executions"*, Computer Systems Laboratory Technical Report CSL–TR–93–578, (PAVG No. 65), September, 1993.

8. Mann, W., Belz, F.C., and Corneil, P., *"A Rapide 1.0 Definition of the ADAGE Avionics System"*, Computer Systems Laboratory Technical Report CSL–TR–93–585, (PAVG No. 66), November, 1993.

9. Katiyar, D., Luckham, D.C., Mitchell J., and Meldal, S., *"Polymorphism and Subtyping in Interfaces."*, in proceedings of ACM Workshop on Interface Definition Languages, pages 22-34, January 20th.,1994.

10. Katiyar, D., Luckham, D.C., and Mitchell J., *"A Type System for Prototyping Languages."*, in proceedings of 21st ACM Symp. on Principles of Programming Languages, page 138-150, January 17-19th., 1994. Also Computer Systems Laboratory Technical Report CSL–TR–94–603, (PAVG No. 67), January, 1994.

11. Luckham, D.C., Augustin, L.M., Kenney, J.J., Vera, J., Bryan, D., and Mann, W., *"Specification and Analysis of System Architecture Using Rapide."*, Computer Systems Laboratory Technical Report CSL–TR–94–608, (PAVG No. 68), April, 1994 and to be appeared in IEEE Transactions on Software Engineering Special Issue on Software Architecture, 1995.

12. Luckham, D.C., Vera, J., *"An Event-Based Architecture Definition Language"*, will be published in IEEE-Transactions on Software Engineering, 1995.

References

[1] Frank Belz and David C. Luckham. A new approach to prototyping Ada–based hardware/software systems. In *Proceedings of the ACM Tri-Ada Conference*, Baltimore, December 1990. ACM Press.

[2] Colin J. Fidge. Timestamps in message-passing systems that preserve the partial ordering. *Australian Computer Science Communications*, 10(1):55–66, February 1988.

[3] Benoit A. Gennart and David C. Luckham. Validating discrete event simulations using event pattern mappings. In *Proceedings of the 29th Design Automation Conference (DAC)*, pages 414–419, Anaheim, CA, June 1992. IEEE Computer Society Press.

[4] D. Katiyar, D. C. Luckham, N. Madhav, S. Meldal, J. C. Mitchell, and S. Sankar. Subtyping, assignment, and cloning in a concurrent object-oriented language. In *Proceedings of the DARPA Software Technology Conference*, pages 458–470, Los Angeles, California, April 1992. DARPA.

[5] John. J. Kenney. Banking on X/Open. To appear as a Stanford University Technical Report, 1993.

[6] David Luckham, James Vera, Doug Bryan, Larry Augustin, and Frank Belz. Partial orderings of event sets and their application to prototyping concurrent timed systems. In *Proceedings*

of the DARPA Software Technology Conference, pages 443–451, Los Angeles, California, April 1992. DARPA.

[7] David C. Luckham and James Vera. Event based concepts and language for system architecture. In *Proceedings of the Workshop on Studies of Software Design*, May 1993.

[8] David C. Luckham, James Vera, Doug Bryan, Larry Augustin, and Frank Belz. Partial orderings of event sets and their application to prototyping concurrent, timed systems. *Journal of Systems and Software*, 21(3):253–265, June 1993.

[9] Sigurd Meldal, Sriram Sankar, and James Vera. Exploiting locality in maintaining potential causality. In *Proceedings of the Tenth Annual ACM Symposium on Principles of Distributed Computing*, pages 231–239, New York, NY, August 1991. ACM Press. Also Stanford University Computer Systems Laboratory Technical Report No. CSL–TR–91–466.

[10] A. Santoro. Case study in prototyping with Rapide: Shared memory multiprocessor system. Technical Report CSL–TR–93–564, Computer Systems Laboratory, Stanford University, March 1993.

[11] X/Open Company Ltd., Apex Plaza, Forbury Road, Reading, Berkshire RGI 1AX, U.K. *Distributed Transaction Processing: Reference Model, Version 2*, November 1993. Guide.